Theoretical Computer Science 83 (1991) 249-260
Elsevier

Nonacceptability criteria and closure properties for the class of languages accepted by binary systolic tree automata

E. Fachini[1]

Dipartimento di Informatica ed Applicazioni, Università di Salerno, 84081 Baronissi, Italy

A. Maggiolo Schettini, G. Resta[2] and D. Sangiorgi[3]

Dipartamento di Informatica, Università di Pisa, 56100 Pisa, Italy

Communicated by A. Salomaa
Received September 1988

Abstract

Fachini, E., A. Maggiolo Schettini, G. Resta and D. Sangiorgi, Nonacceptability criteria and closure properties for the class of languages accepted by binary systolic tree automata, Theoretical Computer Science 83 (1991) 249-260.

In this paper a contribution is given to the solution of the problem of finding an inductive characterization of the class of languages accepted by binary systolic tree automata, $\mathscr{L}$(BSTA), in terms of the closure of a class of languages with respect to certain operations. It is shown that $\mathscr{L}$(BSTA) is closed with respect to some new operations: selective concatenation, restricted concatenation and restricted iteration. The known nonclosure of $\mathscr{L}$(BSTA) with respect to classical language operations, like concatenation and Kleene iteration is proved here by using a new nonacceptability criterion.

1. Introduction

Systolic tree automata have been introduced by Culik II et al. in [4] as a tool to study power, limitations and properties of systolic systems in which the communication structure is a tree. Systolic systems have been introduced by Kung in [8]; they

[1] Present address: Dipartimento Ingegneria Elettronica, II Università di Roma 'Tor Vergata', Via O. Raimondo, 8-00173 Roma, Italy.

[2] Present address: I.E.I.-C.N.R., Via S. Maria, 46-56100 Pisa, Italy.

[3] Present address: Department of Computer Science, University of Edinburgh, Edinburgh EH9 3JZ, UK.

0304-3975/91/$03.50

are arrays of synchronized processors which process data in parallel by transmitting them, in a rythmic fashion, from one processor to the ones to which it is connected.

The new model has given rise to new problems which are also of interest from the point of view of classical language theory. In [2, 3, 5, 7, 9, 10], the class of systolic tree automata where the underlying tree is an infinite binary tree, BSTA, has been investigated and a number of interesting results have been established.

An open problem is to find an inductive characterization of the class of languages accepted by BSTA, $\mathscr{L}$(BSTA), in terms of the closure of a class of languages with respect to certain operations (see [2]).

This paper aims to contribute to the solution of the above mentioned problem. The main result we show here is that $\mathscr{L}$(BSTA) is closed with respect to some new operations: selective concatenation, restricted concatenation and restricted iteration. The operations are defined by imposing some restrictions upon classical concatenation and Kleene iteration. These restrictions are necessary because $\mathscr{L}$(BSTA) is not closed with respect to concatenation and Kleene iteration as Ibarra and Kim proved in [7]. A new and much simpler proof of this result is given here thanks to a new *nonacceptability criterion*.

The results we have given are easily generalized to the case of t-ary STA, i.e. systolic automata with t-ary trees as underlying structure.

The paper is organized as follows: in Section 2 the basic definitions and two nonacceptability criteria are given, Section 3 deals with nonclosure properties and some simple closure results, in Section 4 the closure of $\mathscr{L}$(BSTA) with respect to selective concatenation, restricted concatenation is proved, Section 5 contains the proof of the closure of $\mathscr{L}$(BSTA) with respect to restricted iteration, Section 6 is devoted to some conclusions. In Sections 3, 4, 5 several examples of BSTA acceptable languages are given.

2. Preliminaries and nonacceptability criteria

A deterministic binary systolic tree automaton (BSTA) $K = (\Sigma, Q, F, h, \#)$ consists of an infinite binary leafless tree configuration of identical processors with unit propagation delay between the processors. The set Q, $\Sigma \subseteq Q$ and $F \subseteq Q$, are the operational alphabet, the input alphabet and the set of final states, respectively; $\#$ is a special symbol belonging to Q; each processor computes the function $f: Q \times Q \to Q$, satisfying the condition that $f(x, y) = \#$ iff $x = y = \#$. An input word $w = a_1 \ldots a_n$ is given as input to K as follows: the nodes, in the first level containing a number of nodes greater than or equal to the length of w, receive the letters of w in the order from left to right and the eventually remaining nodes in the level receive $\#$. A processor which receives the input symbol x enters the state x. Now the information flows bottom-up and in parallel. If the sons of a node have already entered the states x and y, respectively, their father enters the state $h(x, y)$. The BSTA K accepts the input word if and only if the root enters a final state. We will call $\mathscr{L}(K)$ the set of all words accepted by K and $\mathscr{L}$(BSTA) the family of BSTA acceptable languages.

Thanks to some "normal form" results, the definition given here is simpler than the original one in [2]. Instead of an unlabelled infinite leafless binary tree, it could be possible to take a labelled one. In this case the function h is replaced by a set of functions h_a where a is one of the node labels. Provided that the considered labelled tree is a regular tree (i.e. it contains only finitely many different infinite subtrees), the class of accepted languages is not affected by this generalization (see [4]). An input function different from the identity is also considered in [2, 3] as well as the possibility that $h(\#, \#) \neq \#$, but also in these cases the class of accepted language does not change, (see [1]).

The model defined here can be made nondeterministic by making the function h multivalued, but also this generalization of the model does not enlarge $\mathscr{L}$(BSTA) (this property will be widely exploited here) [2].

We will denote the jth node in the ith level of the tree underlying the automaton K by $K_{i,j}$. The complement of a language $L \subseteq \Sigma^*$ will be denoted $\sim L$.

We now give two criteria to decide that a language L is not BSTA acceptable. The first appeared in [4] and was later improved in [5]. The second is introduced here; it formalizes and generalizes an argument often used in studies on BSTA to prove nonacceptability.

To state Criterion 2.2 we need the following definition.

Definition 2.1. Given a language L let L_i be the set $\{w \in L \mid 2^{i-1} < |2| \leq 2^i\}$ for $i \in \mathbb{N}$. The language L is called *subexponential* if $\forall n\ \exists k$ such that $|L_k| > n$. A language L has *limited tail ambiguity* iff there exists a natural t such that for every $k \in \mathbb{N}$ and $w \in L$, with $|w| = 2^{k-1}$, there are at most t words w' with $|w'| \leq 2^{k-1}$, such that $ww' \in L$. Given a language L we define $\mathrm{Pr}_k = \{w \mid |w| = 2^k, \exists w'\ 0 < |w'| \leq 2^k, ww' \in L\}$. We say that L contains *words with univocal suffix* if for every k and $w \in \mathrm{Pr}_k$, there exists at least a word w' with $0 < |w'| \leq 2^k$ such that $ww' \in L$ and $vw' \notin L$, where v is a word in Pr_k different from w.

Criterion 2.2. *If a language L is subexponential, with limited tail ambiguity, and contains words with univocal suffix, $L \notin \mathscr{L}$(BSTA).*

Example 2.3. The language $L = \{a^n b^{2m} \mid n + m = k,\ n + m = 2^k,\ n, m, k > 0\}$ is not a BSTA language. Actually, $|L_i| = 2^i - 1$ and therefore L is subexponential. As regards tail ambiguity we have $t = 1$ and the words in L have univocal suffix.

We give now the other criterion, which needs the following definition.

Definition 2.4. Given a language L and $w \in \Sigma^*$, $1 < |w| \leq 2^i$, for $1 \leq i$ we define $\mathrm{Pr}_i^w = \{w' \in \Sigma^{2^i} \mid w'w \in L\}$ and $S_x = \max_{1 \leq |w| \leq 2^x}(|\mathrm{Pr}_x^w|)$. The set Pr_i^w contains those prefixes of length 2^i which, with a suffix w, give a word in L.

Criterion 2.5. *Given a language L, if there exists an infinite subset $A \subseteq \mathbb{N}$ such that for every $n \in A$ there exists $x \in \mathbb{N}$ such that $|\mathrm{Pr}_x|/S_x > n$, then $L \notin \mathscr{L}(\mathrm{BSTA})$.*

Proof. Let us suppose that there exists a BSTA K, with q states, that accepts L. As A is infinite, there will be j such that $|\mathrm{Pr}_j|/S_j > q$. There will be at least $\lceil |\mathrm{Pr}_j| \rceil / q$ of these prefixes, which give the same output. Hence, once we have fixed the suffix, say w, of one of these words, we can complete it in at least $\lceil |\mathrm{Pr}_j| \rceil / q$ ways and obtain different words which must belong to L. But the suffix w might be completed in at most S_j different ways. Therefore we have obtained words accepted by K that cannot belong to L, because $\lceil |\mathrm{Pr}_j| \rceil / q > S_j$. □

Example 2.6. Let L be the language $\{a^{2^n} b^{2^m} \mid n, m \geq 1\}$. L has not limited tail ambiguity, so we cannot apply criterion 2.2. It holds that $|\mathrm{Pr}_x| = x - 1$, as the words of the set $\{a^{2^j} b^{(2^{x-1}-2^j)} \mid 1 < j \leq x\}$ are the possible prefixes. Besides, $S_x = 2$ because a given word has at most two prefixes. It is clear that $|\mathrm{Pr}_x|/S_x$ grows arbitrarily, and therefore $L \notin \mathscr{L}(\mathrm{BSTA})$.

Example 2.7. Let L be the language $\{a\}^*\{b^{2^n} \mid n \geq 0\}$; L has not limited tail ambiguity, so we cannot apply Criterion 2.2. It holds that $|\mathrm{Pr}_x| = 2^x$ and $S_x = x$, so that L does not belong to $\mathscr{L}(\mathrm{BSTA})$.

3. Closure of $\mathscr{L}$(BSTA) with respect to classical language operations

In this section we recall some closure results of $\mathscr{L}(\mathrm{BSTA})$ and we prove that $\mathscr{L}(\mathrm{BSTA})$ is not closed with respect to homomorphism. We also prove some simple closure results which will be useful in the next sections.

Theorem 3.1. (a) $\mathscr{L}(\mathrm{BSTA})$ *is closed with respect to boolean operations and right concatenation with regular sets.*

(b) $\mathscr{L}(\mathrm{BSTA})$ *is not closed with respect to left concatenation with regular sets, reversal, inverse homomorphism and Kleene plus operator.*

Proof. For (a) see [2, 7]. A proof of (b) is given by Ibarra and Kim in [7] and it is based on the fact that the language L of our Example 2.7 does not belong to $\mathscr{L}(\mathrm{BSTA})$. Note that the argument given in [7] to prove that L does not belong to $\mathscr{L}(\mathrm{BSTA})$ is much more complicated than the one used here. □

Theorem 3.2. $\mathscr{L}(\mathrm{BSTA})$ *is not closed with respect to homomorphism.*

Proof. Take the language $L = \{a^n b^m \mid n + m = 2^k, n, m, k \geq 0\} \in \mathscr{L}(\mathrm{BSTA})$. Consider the homomorphism $h : \{a, b\}^* \to \{a, b\}^*$ such that $h(a) = a$ and $h(b) = b^2$. It holds that $h(L) \notin \mathscr{L}(\mathrm{BSTA})$. Actually, $h(L)$ is the language of Example 2.3. □

This result is a particular case of Theorem 4.5 of [1] about the nonclosure with respect to homomorphism of the class of systolic tree automata whose underlying tree is an initial prefix tree (i.e. a tree such that all the subtrees rooted in the nodes in the leftmost path are equal). The result of Theorem 3.2 has been proved independently by Pardubska [10].

Definition 3.3. Given an alphabet Σ, a word $w \in \Sigma^*$ and a letter $v \in \Sigma$, we define the *left append function*: $A_v : \Sigma^* \to \Sigma^*$ as $A_v(w) = vw$. Taking $L \subseteq \Sigma^*$, let $A_v(L) = \{A_v(w) \mid w \in L\}$.

Lemma 3.4. *The class* $\mathscr{L}$(BSTA) *is closed with respect to* A_v, *for any letter* v.

Proof. Let $K = (\Sigma, Q, F, f, \not\#)$ be a BSTA with $\mathscr{L}(K) = L$. We shall construct a nondeterministic BSTA $\bar{K} = (\Sigma, \bar{Q}, \bar{F}, \bar{f}, \not\#)$ such that $\mathscr{L}(\bar{K}) = \bar{L} = A_v(L)$.

Let us take $w \in L$ and $\bar{w} = vw \in \bar{L}$. First, we suppose that w and $\bar{w}$ are given as input to K and $\bar{K}$ in the same level, say the kth one. Each processor in the $(k-1)$th level, by receiving v_1, v_2 as input values, nondeterministically will enter a three component state (v_1, s, v_2) where $s = f(v_2, v_3)$ is the state entered by the corresponding processor of K, which receives v_2, v_3 as input values. In the next computation steps each processor, by receiving (v_1, s_1, v_2) and (v_3, s_2, v_4) as input values, enters the state $(v_1, f(s_1, s_2), v_4)$ if $v_2 = v_3$ and a rejecting state otherwise.

The final states are of the form $(v, s, \not\#)$ with $s \in F$. In the case $\bar{w}$ is given as input in the $(k+1)$th level because $|w| = 2^k$, the behaviour of the processors in the left subtree of the root is the same as in the previous case, but the processor $\bar{K}_{i,j}$ simulates the processor $K_{i-1,j}$. Only the last letter v' of $\bar{w}$ is given as input to the right subtree of the root. This letter v' is transmitted to the root of $\bar{K}$. Now if we denote output$(\bar{K}_{1,1}, \bar{w})$ by (v_1, s_1, v_2), the root must check only that $v_2 = v'$ and if this holds, it outputs s. □

Corollary 3.5. *The class* $\mathscr{L}$(BSTA) *is closed with respect to left concatenation with a finite set.*

Proof. Let F be the considered finite set of words over an alphabet Σ. For any $w = x_1 \ldots x_k$, $x_i \in \Sigma$ for $1 \leq i \leq k$, let $A_w(u) = A_{x_1}(\ldots(A_{x_k}(u))\ldots)$, $u \in \Sigma^*$, and $A_w(L) = \{A_w(u) \mid u \in L\}$. Then it holds that $F \cdot L = \bigcup_{w \in F} A_w(L)$. □

Definition 3.6. Let us define the *cut operations* C^+ and C^- as follows:

$$C^+(\lambda) = C^-(\lambda) = \lambda;$$

$$C^+(xa) = x \quad \text{and} \quad C^-(ax) = x \quad \text{for every } x \in \Sigma^*, a \in \Sigma.$$

Lemma 3.7. *The class* $\mathscr{L}$(BSTA) *is closed with respect to* C^+ *and* C^-.

Proof. The proofs for C^+ and C^- are conceptually similar to that of Lemma 3.4. □

Corollary 3.8. *The class* $\mathscr{L}$(BSTA) *is closed with respect to left and right quotient with finite sets.*

Proof. Immediate. □

4. Selective and restricted concatenation

We give here restrictions on the usual concatenation, to obtain operations which do not lead out of $\mathscr{L}$(BSTA).

The problem is that the recognition of nonregular languages is strongly related to the structure of the underlying tree. In fact, from the definition of STA, a word w enters aligned on the left of the tree and therefore there is a well defined correspondence between letters of the word and structure of the tree. This correspondence gets lost if another word is concatenated to the left of w. In the t-ary trees, and in particular in the binary trees studied here, we shall overcome this problem by limiting the length of a word with respect to the length of the word on the left in the concatenation. For this purpose we shall use the height constants in the operations we define. We do not lose the BSTA-acceptability if, beginning from a situation of this kind, every word of the concatenation is shifted a fixed number of positions. This possibility of shifting is given by the shift constants.

We have called the operations obtained in this manner, selective concatenation (SC) and restricted concatenation (RC). The operations SC and RC differ by the conditions related to the height constants.

Definition 4.1. For given alphabets Σ_i, let it be $L_i \subseteq \Sigma_i^*$, $\Sigma = \bigcup_{1 \leqslant j \leqslant m} \Sigma_j$, $h_i, k_i \in \mathbb{Z}$, for $1 \leqslant i \leqslant m$. We obtain L by *selective concatenation* with respect to $h_1, \ldots, h_m$ (shift constants), $k_1, \ldots, k_{m-1}$ (height constants) from $L_1, \ldots, L_m$ (briefly $L = \mathrm{SC}_{k_1, \ldots, k_{m-1}}^{h_1, \ldots, h_m}(L_1, \ldots, L_m)$) if

$$L = \{w_1 \ldots w_m \in \Sigma^* \mid |w_i| = 2^{n_i} + h_i,\ w_i \in L_i,\ 1 \leqslant i \leqslant m \text{ and } n_j \geqslant n_{j+1} + k_j,\ 1 \leqslant j < m\}.$$

Example 4.2. Taken $L_1 = \{a\}^*$, $L_2 = \{b\}^*$, $L_3 = \{c\}^*$, it results that

$$\mathrm{SC}_{2,-7}^{3,-1,5}(L_1, L_2, L_3) = \{a^{2^{n_1}+3} b^{2^{n_2}-1} c^{2^{n_3}+5} \mid n_1 \geqslant n_2 + 2,\ n_2 \geqslant n_3 - 7\}.$$

In this example we see that by the SC operation first the words of length $2^n + h_i$ are selected from every language L_i and then concatenated respecting the conditions expressed by the height constants.

Note that the selective concatenation of m languages does not give the same result as the repeated application of selective concatenation of two languages.

We give now a definition that we shall often use in the following.

Definition 4.3. Let z be a word over the alphabet Σ, which is given as input to the BSTA K in the level k and let $T_1, \ldots, T_m$ be $m \geq 1$ subtrees of the underlying tree of K whose heights, from the level k, are $h_1, \ldots, h_m$ respectively; let $n_0 = 0$ and $n_i = \sum_{j=1}^{i} 2^{h_j}$. Suppose that $N(k, r+n_{i-1}+j_i)$ is the j_ith leaf of T_i for $1 \leq j_i \leq 2^{h_i}$, $1 \leq i \leq m$, $r \geq 0$. We say that the subword x of z, with $x = x_1 \ldots x_t$, $x_i \in \Sigma$ for $1 \leq j \leq t$, *fills exactly* the subtrees $T_1, \ldots, T_m$ if $t = n_m$ and, when z is given as input word to K, x_j is the input value for $N(k, r+j)$ for $1 \leq j \leq t$.

Example 4.4. Consider the tree in Fig. 1. Here v_1 fills exactly T_1, v_2 fills exactly T_1, $T_2(r = 2^{h_1})$, v_3 fills exactly T_3, $T_4(r = 2^{h_1+2})$.

Theorem 4.5. *The class $\mathscr{L}$(BSTA) is closed with respect to selective concatenation.*

Proof. Let L be defined as $\mathrm{SC}^{h_1,\ldots,h_m}_{k_1,\ldots,k_{m-1}}(L_1, \ldots, L_m)$ with $L_i = \mathscr{L}(K_i)$ for a BSTA K_i, $1 \leq i \leq m$. We shall describe the behavior of a BSTA K accepting L. In the following when we use the letter i we shall always mean that $1 \leq i \leq m$. We shall denote by w a word of L, with $w = w_1 \ldots w_m$, $w_i \in L_i$.

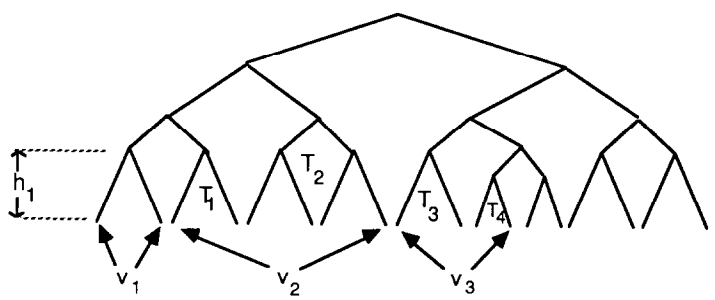

Fig. 1.

First we deal with the case when all the shift constants are equal to zero. For every word w_i we must check that $w_i \in L_i$ and the length of w_i is related to the length of w_{i+1} by the proper height constant. The controls are easy if all the height constants are positive, because every w_i exactly fills a subtree and these subtrees have decreasing heights. We are not in this situation if at least one k_i is negative. To deal with this case, let us introduce the following definition.

We call $S_i = (S_{i,1}, S_{i,2}, \ldots, S_{i,j_i})$ the tuple of minimal length consisting of subtrees of the same height, filled exactly by w_i. Let us denote the root of $S_{i,k}$ by $r_{i,k}$. Let N_i be the first node from the bottom which is an ancestor of all the roots of the subtrees in S_i and S_{i+1} (S_i only if $i = m$).

We shall show by induction on m that there exists an upperbound $M_{k_1,\ldots,k_{m-1}}$ on the number d_i of levels between the level of $r_{i,k}$, $k \in \{1,\ldots,j_i\}$, and the level of N_i.

If $m=1$ it is obviously true. Let us suppose now it is true for every $m < \bar{m}$, we show that it holds for $\bar{m}$. First we define the property $\mathcal{P}(i)$, for $1 \leq i < m$, as follows:

$$\mathcal{P}(i) \equiv n_i \geq n_{i+1} + \bar{m} \cdot (2 \cdot \bar{k} + 1) \quad \text{where } \bar{k} = \max_{1 \leq j \leq m}(|k_j|).$$

When $\mathcal{P}(i)$ is true we have the following situation. If $K_{p,q}$ is r_{i,j_i}, then $K_{p,q+1}$ is an ancestor of all the roots of subtrees in $S_{i+1},\ldots,S_{\bar{m}}$. This may happen if the height, $h(i)$, of the subtree with root $K_{p,q+1}$ (i.e. the height of the subtrees of S_i) is greater than $\lceil(\log_2(|w_{i+1}\ldots w_{\bar{m}}|)\rceil$. It holds $h(i) \geq \min_{1 \leq t \leq i}(n_t)$. Now $n_{i-1} \geq n_i + k_{i-1} \geq n_i - \bar{k}$ and also $n_j \geq n_i - (i-j)\cdot\bar{k}$ for $1 \leq j \leq i-1$. Hence $h(i) \geq n_i - (i-j)\cdot\bar{k} \geq n_i - \bar{m}\cdot\bar{k}$. On the other hand with a computation similar to the previous one, we obtain that $|w_j| \leq 2^{n_{i+1}+\bar{m}\cdot\bar{k}}$, for $i+1 \leq j \leq \bar{m}$ and hence

$$\begin{aligned}\lceil(\log_2(|w_{i+1}\ldots w_{\bar{m}}|)\rceil &\leq \lceil(\log_2((\bar{m}-i)\cdot 2^{n_{i+1}+\bar{m}\cdot\bar{k}})\rceil \\ &\leq \bar{m} + n_{i+1} + \bar{m}\cdot\bar{k} = n_{i+1} + \bar{m}\cdot(\bar{k}+1).\end{aligned}$$

So we can conclude that $h(i) \geq \lceil(\log_2(|w_{i+1}\ldots w_{\bar{m}}|)\rceil$ if $n_i - \bar{m}\cdot\bar{k} \geq n_{i+1} + \bar{m}\cdot(\bar{k}+1)$ and this condition is equivalent to $\mathcal{P}(i)$.

Let us distinguish two cases:

Case 1: $\nexists j$ *such that* $\mathcal{P}(j)$ *is true.* From the relations $n_i \geq n_{i+1} + k_i$ and not $\mathcal{P}(i)$, it results that $n_i - \bar{m}\cdot(2\cdot\bar{k}+1) < n_{i+1} \leq n_i - k_i$; consequently, once we have fixed n_1, we have a finite number of n-tuples $(n_1,\ldots,n_{\bar{m}})$ belonging to Case 1. Given a $\bar{m}$-uple $(n_1,\ldots,n_{\bar{m}})$, by adding the same constant c to each component n_i, the relative $\bar{m}$-uple $(d_1,\ldots,d_{\bar{m}})$ does not change (for d_i defined as above). Hence, we have only a finite number of different $\bar{m}$-tuples $(d_1,\ldots,d_{\bar{m}})$ and it is trivial to find an upperbound $\bar{M}_{k_1,\ldots,k_{m-1}}$.

Case 2: $\exists j$ *such that* $\mathcal{P}(j)$ *is true.* Let $\bar{j}$ be $\min j$ such that $\mathcal{P}(j)$ is true. For $j \leq \bar{j}$, an upperbound $M'_{k_1,\ldots,k_{\bar{j}}}$ to d_j can be found by means of an argument similar to that of Case 1. For $j > \bar{j}$, as $\mathcal{P}(j)$ is supposed to be true, there is a subtree filled exactly by $w_{\bar{j}+1}\ldots w_{\bar{m}}z$, $z \in \#^*$. Therefore, by exploiting the induction hypothesis an upperbound is given by $M_{k_{\bar{j}+1},\ldots,k_{\bar{m}-1}}$. We can now take

$$M''_{k_1,\ldots,k_{\bar{m}-1}} = \max_{1 \leq j < \bar{m}}(\max(M'_{k_1,\ldots,k_j}, M_{k_{j+1},\ldots,k_{\bar{m}-1}})).$$

Finally we can assign the value $\max(M''_{k_1,\ldots,k_{\bar{m}-1}}, \bar{M}_{k_1,\ldots,k_{\bar{m}-1}})$ to $M_{k_1,\ldots,k_{\bar{m}-1}}$. There is an upperbound also to the cardinality of S_i. This cardinality will be at most $2^{(n_i - n_{\bar{j}})}$ where $n_{\bar{j}} = \min_{j \leq i}(n_j)$. But, as we have already shown, $\min_{1 \leq t \leq i} n_t \geq n_i - m\cdot\bar{k}$. Hence $n_i - n_{\bar{j}} \leq m \cdot k$, and $2^{m\cdot\bar{k}}$ is the upperbound we were looking for.

We now describe the behaviour of K. Each input processor of K nondeterministically guesses to which w_i the letter it receives belongs. Then we simulate the computation of a BSTA K_i accepting L_i on the subtrees S_i. Each processor guesses the position of the nodes $r_{i,j}$, $1 \leq j \leq j_i$. In these nodes a counter is set to 0 and subsequently it is increased at every level. These counters will be used in the nodes N_i to check that the nodes $r_{i,j}$ are in the same level. At the same time it is also

checked that j_i is a power of 2. Moreover, by means of the counters, we shall check the condition $n_i \geq n_{i+1} + k_i$. In the nodes N_i we also complete the simulation of K_i on w_i from the partial computation obtained in the nodes $r_{i,j}$, transmitted unchanged until N_i. It should be clear that these controls are possible thanks to the upperbounds for the value of d_i and j_i that we have proved above.

Let us now consider the case where at least one constant h_i is not 0; in this case the construction given before does not work, because d_i and j_i are not limitable.

We shall use the following notation:

$$ {}_{q_1}L_{q_2} = \begin{cases} (C^+)^{q_2}((C^-)^{q_1}(L)) & \text{if } q_1 < 0,\ q_2 < 0, \\ ((C^-)^{q_1}(L)) \cdot \Sigma^{q_2} & \text{if } q_1 < 0,\ q_2 \geq 0, \\ (C^+)^{q_2}(\Sigma^{q_1} \cdot L) & \text{if } q_1 \geq 0,\ q_2 < 0, \\ (\Sigma^{q_1} \cdot L) \cdot \Sigma^{q_2} & \text{if } q_1 \geq 0,\ q_2 \geq 0. \end{cases} $$

As we can observe, ${}_{q_1}L_{q_2}$ is defined through operations which do not lead out of $\mathscr{L}$(BSTA).

Let us now define the constants $\bar{h}_i$: $\bar{h}_0 = 0$, $\bar{h}_i = \sum_{k=1}^{i} h_i$. We consider the languages $\bar{L}_i = {}_{\bar{h}_{i-1}}(L_i)_{-\bar{h}_i}$; $\bar{L}_i = \mathscr{L}(\bar{K}_i)$ for a BSTA $\bar{K}_i$. Let $\bar{L}$ be $\mathrm{SC}^{0,\ldots,0}_{k_1,\ldots,k_{m-1}}(\bar{L}_1, \ldots, \bar{L}_m)$; as the shift constants are equal to 0, we already know that $\bar{L}$ is accepted by a BSTA $\bar{K}$ constructed following the first part of this proof. In general it holds that $\bar{L} \neq L$. In fact, let us suppose that, for example, $\bar{h}_1 > 0$. This means that to construct the words in $\bar{L}_1$ and $\bar{L}_2$, we must cut the last $\bar{h}_1$ letters from any word in L_1, and add $\bar{h}_1$ letters to the beginning of any word of L_2. But the string that is cut may be different from the one which is added and hence there may be words in $\bar{L}$ that do not belong to L.

It is easy to construct a BSTA accepting L, starting from a BSTA that accepts $\bar{L}$. Having always in mind the considered case of $\bar{h}_1 > 0$, the BSTA $\bar{K}_1$ constructed from K_1, must guess the last $\bar{h}_1$ letters of w_1 that it does not receive; besides that, $\bar{K}_2$ transmits towards the root the first $\bar{h}_1$ letters of its input for the next equality control. With analogous modifications for every $\bar{h}_i$, we obtain a BSTA accepting L, hence the thesis. □

In this definition of SC we have conditions of the form $n_i \geq n_{i+1} + k_i$, but we can change them in equality conditions like $n_i = n_{i+1} + k_i$, by exploiting boolean operations.

Example 4.6. The language $L' = \{a^{2^{n_1}} b^{2^{n_2}} \mid n_1 = n_2 + 3\}$ may be obtained as follows: $L' = \mathrm{SC}^{0,0}_{+3}(L_1, L_2) \cap (\sim \mathrm{SC}^{0,0}_{+4}(L_1, L_2))$.

Definition 4.7. For an alphabet Σ, $h \in \mathbb{Z}$, $k \in \mathbb{N} - \{0\}$, let $\mathscr{R}^h_k \subseteq (\Sigma^*)^2$ be the relation such that given v_1, $v_2 \in \Sigma^*$, with $|v_1| - h = 2^{n_1} + \cdots + 2^{n_r}$, $|v_2| = 2^{m_1} + \cdots + 2^{m_s}$ for $n_1 > \cdots > n_r$, $m_1 > \cdots > m_s$, $r, s \geq 1$, it results that $\mathscr{R}^h_k(v_1, v_2)$ iff $n_r \geq m_1 + k$.

Let Σ_i be alphabets with $L_i \subseteq \Sigma_i^*$ and such that $\bigcup_{1 \leq j \leq m} \Sigma_j = \Sigma$, $1 \leq i \leq 2$. We say that L is obtained from L_1, L_2 by *restricted concatenation* with respect to h, k (shortly $L = \mathrm{RC}^h_k(L_1, L_2)$) if $L = \{w_1 w_2 \in \Sigma^* \mid w_i \in L_i$ and $\mathscr{R}^h_k(w_1, w_2)$, $i = 1, 2\}$.

When it is not necessary, we shall omit the reference to h and k.

Example 4.8. Taking

$$L_1 = \{a^{2^{n+3}}b^{2^n} \mid n \in \mathbb{N}\}, \qquad L_2 = \{a^{2^{n_1}} \dots a^{2^{n_m}} \mid n_j = n_{j+1}+3,\ 1 \leq j < m\},$$

$$\mathrm{RC}_7^2(L_1 cc, L_2) = \{a^{2^{n+3}}b^{2^n}cca^{2^{n_1}} \dots a^{2^{n_m}} \mid m > 1,\ n_j = n_{j+1}+3,\ 1 \leq j < m,\ n \geq n_1+7\}.$$

Note that $L_1, L_2 \in \mathscr{L}(\mathrm{BSTA})$ (see Examples 4.6 and 5.4).

Theorem 4.9. *The class $\mathscr{L}$(BSTA) is closed with respect to restricted concatenation.*

Proof. Let $L \subseteq \Sigma^*$ be obtained as $\mathrm{RC}_k^h(L_1, L_2)$ for $L_1, L_2 \subseteq \mathscr{L}(\mathrm{BSTA})$; let $w = w_1w_2$ be in L, $w_1 \in L_1$, $w_2 \in L_2$. We describe the behaviour of a nondeterministic automaton H which accepts L.

First let us suppose that $h = 0$. We call K_i a BSTA such that $\mathscr{L}(K_i) = L_i$; moreover, let S_i the smallest subtree of H filled exactly by $z = w_iu$ for $u \in \Sigma^*\{\#\}^*$. Each input processor must guess whether the letter it receives belongs to either w_1 or w_2. The root of S_i must simulate the root of the underlying tree of K_i.

The other condition which must be verified is $\mathscr{R}_k^0(w_1, w_2)$. We write $|w_i|$ as the sum of strictly decreasing powers of 2: $|w_i| = 2^{n_1} + \dots + 2^{n_{q_i}}$. Then it will be possible to locate q_i subtrees $P_{i,1}, \dots, P_{i,q_i}$ of S_i whose heights (from the input level) are $n_1, \dots, n_{q_i}$, respectively, and such that w_i fills them exactly. The rules of H must allow us to check whether there is a difference of at least k levels between the roots of P_{1,q_1} and $P_{2,1}$; this is in fact the meaning of $\mathscr{R}_k^0(w_1, w_2)$.

Let us suppose now that $h \neq 0$. The problem that we have is similar to the one in the proof of Theorem 3.1, about the closure of $\mathscr{L}$(BSTA) with respect to SC in the case where not all the shift constants are equal to 0. The solution is analogous. □

The condition on the height constant can be changed into equality in a way analogous to the one seen for SC operation.

5. Restricted iteration

The class $\mathscr{L}$(BSTA) is not closed with respect to iteration (for instance the language $L = \{a^{2^{n-1}}b \mid n \in \mathbb{N}\}$ is in $\mathscr{L}$(BSTA), but L^* is not, as one may prove by exploiting Criterion 2.5). The reason is essentially the same for which $\mathscr{L}$(BSTA) is not closed under concatenation. Therefore we may introduce restrictions which are similar to the ones in the restricted concatenation. There is no shift constant; actually if we had a shift constant h, as the number m of words which are concatenated in an iteration is variable, the total shift of the last word of the iteration would be $(m-1) \cdot h$ and hence arbitrarily big.

Definition 5.1. Let Σ be an alphabet with $L \subseteq \Sigma^*$ and $k \in \mathbb{N} - \{0\}$. We say that L' is obtained from L by *restricted iteration* with respect to k (shortly $L' = \mathrm{RI}_k(L)$) if

$$L' = \{w_1 \dots w_m \in \Sigma^* \mid w_i \in L,\ 1 \leq i \leq m,\ \mathscr{R}_k^0(w_j, w_{j+1}) \text{ for } 1 \leq j < m\}.$$

Example 5.2. Given $L_1 = \{a^{2^n} \mid n \geq 0\}$, $\mathrm{RI}_3(L_1) = \{a^{2^{n_1}} \dots a^{2^{n_m}} \mid m > 1,\ n_j \geq n_{j+1} + 3,\ 1 \leq j \leq m\}$.

Theorem 5.3. *The class $\mathscr{L}$(BSTA) is closed with respect to restricted iteration.*

Proof. The proof is analogous to the one given for the operation RC in the case of shift constant equal to 0. The difference is that now we have to define m subtrees S_i and we have to control $m-1$ relations $\mathscr{R}_k^0(w_i, w_{i+1})$, where m is not fixed; all this can be done easily. □

As regards SC and RC operations, we may also change the inequality conditions into equality conditions in RI operation.

Example 5.4. We show now how $L = \{a^{s^{n_1}} \dots a^{2^{n_m}} \mid m > 1,\ n_j = n_{j+1} + 3,\ 1 \leq j \leq m\}$ can be obtained from $L_1 = \{a^{2^n} \mid n \geq 0\}$. Starting from $L_2 = \mathrm{RC}_3^0(L_1, L_1)$ and $L_3 = \mathrm{RC}_4^0(L_1, L_1)$, we obtain

$$L_4 = L_2 \cap (\sim L_3) = \{a^{2^{n_1}} a^{2^{n_2}} \mid n_1 = n_2 + 3\}.$$

Then we have

$$L_5 = \mathrm{RI}_3(L_4) = \{a^{2^{n_1+3}} a^{2^{n_1}} a^{2^{n_2+3}} a^{2^{n_2}} \dots a^{2^{n_m+3}} a^{2^{n_m}} \mid m > 1,\ n_j \geq n_{j+1} + 6,\ 1 \leq j \leq m\},$$

$$L_6 = \mathrm{RC}_3^0(L_1, L_5) = \{a^{2^p} a^{2^{n_1+3}} a^{2^{n_1}} a^{2^{n_2+3}} a^{2^{n_2}} \dots a^{2^{n_m+3}} a^{2^{n_m}} \mid m > 1,\ n_j \geq n_{j+1} + 6,\ 1 \leq j \leq m,\ p \geq n_1 + 6\}.$$

Finally $L = L_5 \cap L_6$.

The scheme used in the example can be always applied.

6. Conclusion

We have used two nonacceptability criteria to show that $\mathscr{L}$(BSTA) is not closed with respect to classical language operations, like concatenation, Kleene iteration, and homomorphism. One of these criteria is new and allows a simpler proof of some known results. We have introduced some new language operations, i.e. selective concantenation, restricted concatenation and restricted iteration with respect to which $\mathscr{L}$(BSTA) is closed.

In [3] it is shown that any language in $\mathscr{L}$(BSTA) can be obtained as the homomorphic image of a binary and suffix E0L language. A binary E0L language is generated by an E0L system whose productions have right-hand sides of length two. A suffix E0L language has words containing $\#^n$, where $\#$ is a special symbol, only as a suffix. The homomorphism is the identity on the terminal symbols and deletes the special symbol $\#$.

It is an open problem whether $\mathscr{L}$(BSTA) is the closure of the class of binary E0L languages with respect to boolean operations, right concatenation with regular sets and selective concatenation, restricted concatenation and restricted iteration. We have not found in the literature and we have not been able to give ourselves languages which are BSTA acceptable and do not belong to the previous closure.

Such a characterization would be a useful tool in the design and analysis of systolic systems with a binary tree as communication structure.

Note (added in proof)

Recently an inductive characterisation of $\mathscr{L}$(BSTA) was given by the first author and A. Monti, using similar restricted concatenation and iteration operators plus union.

References

[1] D. Avallone, Automi sistolici con struttura ad albero ed a C-albero, Laurea Thesis, University of Salerno, 1986.

[2] K. Culik II, J. Gruska and A. Salomaa, Systolic automata for VLSI on balanced tree, *Acta Inform.* **18** (1983) 335-344.

[3] K. Culik II, J. Gruska and A. Salomaa, On a family of L languages resulting from systolic tree automata, *Theoret. Comput. Sci.* **23** (1983) 231-242.

[4] K. Culik II, A. Salomaa and D. Wood, Systolic tree acceptors, *RAIRO Inform. Théor.* **18** (1984) 53-69.

[5] E. Fachini and L. Iania, A note on the paper "Systolic tree acceptors" by K. Culik II, A Salomaa and D. Wood, *EATCS Bull.* **28** (1986) 26-30.

[6] J. Gruska, Systolic automata: power, characterizations, nonhomogeneity, in: *Proc. MCFS '84*, Lecture Notes in Computer Science **176** (Springer, Berlin, 1984) 32-49.

[7] O.H. Ibarra and S.M. Kim, A characterization of systolic binary tree automata and applications, *Acta Inform.* **21** (1984) 193-207.

[8] H.T. Kung, Why systolic architecture?, *Comput. Mag.* **15** (1982) 37-46.

[9] M. Patterson, Solution to P8, *EATCS Bull.* **18** (1982) 29.

[10] D. Pardubska, Closure properties of the family of languages defined by systolic tree automata, *Comput. Artificial Intelligence* 7 (1988) 59-64.

[11] G. Rozenberg and A. Salomaa, *The Mathematical Theory of L Systems* (Academic Press, New York, 1980).